Object Tracking

El Mansouri Imad

January 28, 2024

1 Main Class

1.1 Frame

The Frame class represents a single frame within a video sequence in the context of object tracking. It acts as a container for all the active tracks within that particular frame, managing their states. The class is designed to keep track of all objects being monitored in the frame, handling tasks such as adding new tracks, updating existing ones, and retrieving active tracks. One of the key functionalities of the Frame class is the management of track identifiers, ensuring that each track is uniquely and consistently identified across the frames. This is achieved with the help of the IDManager class, which generates new IDs as tracks are added. The ability to add and update tracks within a frame allows for dynamic tracking, accommodating the entry of new objects into the scene and changes in the state of existing objects.

1.2 Track

The Track class is a cornerstone in the object tracking system, representing an individual object being tracked over time. Each instance of this class encapsulates all the necessary information and functionalities related to a single track, such as its unique identifier, bounding box detection, and the state of the track. A particularly notable feature of this class is its optional use of a Kalman filter. When enabled, the Kalman filter provides predictive tracking capabilities, enhancing the accuracy of the track's position in cases of occlusion or other tracking challenges. Additionally, the track can use a model to add visual information to the detection with the integratation of a deep learning-based appearance embeddings of the model.

2 Methodes

2.1 Greedy

The function operates on a greedy approach, selecting the track with the highest IoU for each detection. This method is straightforward and computationally efficient, making it suitable for real-time tracking applications. A threshold (sigma_iou) is used to determine if a detection should be associated with an existing track. If the highest IoU for a detection exceeds this threshold, it suggests a strong match, and the corresponding track is updated with the new detection data. If no track meets this criterion, it implies the detection is of a new object, prompting the creation of a new track. The function optionally considers Kalman filter predictions for calculating IoU. When enabled, the predicted position of the track (rather than the last known position) is used for IoU calculations, enhancing accuracy especially in cases where objects move predictably.

2.2 Hungarian

A fundamental step in this process is the creation of a cost matrix, which quantifies the "cost" of assigning each detection to each track. This matrix is populated based on the IoU (Intersection over Union) metric, a standard measure in object tracking that assesses the overlap between detection and track bounding boxes. The function optionally considers Kalman filter predictions for calculating IoU.

The Hungarian algorithm, a classic optimization approach, is then applied to this cost matrix. The algorithm seeks to minimize the total cost of assignments, effectively finding the most efficient way to match detections to existing tracks. This process is particularly advantageous in scenarios with multiple objects and potential detection-to-track ambiguities.

The function incorporates an IoU threshold (sigma_iou) to guide the assignment process. If the IoU score for a detection-track pair exceeds this threshold, it signifies a strong match, and the detection is assigned to that track. Conversely, detections with no sufficiently high IoU scores are considered to be new objects, leading to the creation of new tracks. Detections: Post-assignment, the function also manages detections that have not been matched to any track. This is a crucial step to ensure that new objects entering the scene are recognized and tracked.

2.3 Resnet

For each detection and track pair, the function calculates both the IoU and the cosine similarity of feature embeddings. IoU assesses the spatial overlap between the detection and track bounding boxes, while cosine similarity measures how similar the appearance features are. These two metrics are then combined into a single score using a weighted approach, where alpha determines the relative importance of the IoU in the score. If the kalman fitler is selected, the used the prediction for both the iou and the cosine similarity.

The combined scores are used to populate the cost matrix. Once the cost matrix is populated, the function employs the Hungarian algorithm to find the optimal assignment of detections to tracks.

2.4 Yolo

Same methode as the Hungarian but use the YOLO pedestrian detection instead of the given one.

3 Visualisation

I have added some visualistaion functions that help for debuggin and get empiric idea of the efficienty of the tracker.

3.1 visualisation

Draws bounding boxes around detected objects in each frame and saves the annotated images. (also draw the prediction of the kalman filter the center of the bounding box)

3.2 create_video

Creates a video from a sequence of image frames stored in a specified directory (i.e the directory created by the visualisation function)

4 Results

I did some hyper-parametrization for the value of the sigma_iou (or threshold). I focused on HOTA, IDF1, IDSW, MT, PT and ML.

HOTA: MPNTrack-pedestrian	HOTA	DetA	AssA	DetRe	DetPr	AssRe	AssPr	LocA	OWTA	HOTA(0)	LocA(0)	HOTALocA(0)					
CLEAR: MPNTrack-pedestrian	MOTA	MOTP	MODA	CLR_Re	CLR_Pr	MTR	PTR	MLR	sMOTA	CLR_TP	CLR_FN	CLR_FP	IDSW	MT	PT	ML	Frag
Identity: MPNTrack-pedestrian	IDF1	IDR	IDP	IDTP	IDFN	IDFP											
Count: MPNTrack-pedestrian	Dets	GT_Dets	IDs	GT_IDs													

4.1 greedy

Methode.GREEDY - [Kalman = False] -[$\sigma = 0.8$]

Method.GREEDY - [Kalman = False] -[σ = 0.8]																	
ADL-Rundle-6	16.05	28.256	9.4816	37.57	45.173	9.9871	65.346	73.787	18.643	23.822	59.619	14.202					
ADL-Rundle-6	2.5754	70.847	9.3831	46.277	55.641	0	91.667	8.3333	-10.915	2318	2691	1848	341	0	22	2	403
ADL-Rundle-6	18.768	17.189	20.667	861	4148	3305											
ADL-Rundle-6	4166	5009	966	24													

4.2 hungarian

Methode.HUNGARIAN - [Kalman = False] -[$\sigma = 0.25$]

Methode.HUNGARIAN - [Kalman = False] -[σ = 0.25]																	
ADL-Rundle-6	17.365	31.308	9.8852	43.873	44.83	10.584	52.674	74.567	20.678	26.464	61.147	16.182					
ADL-Rundle-6	5.5899	71.243	11.779	54.821	56.018	0	91.667	8.3333	-10.175	2746	2263	2156	310	0	22	2	300
ADL-Rundle-6	19.332	19.126	19.543	958	4051	3944											
ADL-Rundle-6	4902	5009	551	24													

4.3 RESNET

No optimisation for RESNET due to its computation time. (10min) Method.RESNET - [Kalman = False] -[$\sigma = 0.25$]

Method.RESNET - [Kalman = False] -[σ = 0.25]																	
ADL-Rundle-6	17.609	30.868	10.281	43.38	44.327	11.323	48.873	74.366	20.989	27.733	60.62	16.812					
ADL-Rundle-6	5.6099	71.113	11.659	54.761	55.957	4.1667	87.5	8.3333	-10.209	2743	2266	2159	303	1	21	2	303
ADL-Rundle-6	19.756	19.545	19.971	979	4030	3923											
ADL-Rundle-6	4902	5009	439	24													

4.4 YOLO

Method.YOLO - [Kalman = False] -[$\sigma = 0.5$]

Method.YOLO - [Kalman = False] -[σ = 0.5]																	
ADL-Rundle-6	39.638	45.749	35.1	50.588	71.885	38.528	78.738	81.268	41.892	49.624	76.658	38.041					
ADL-Rundle-6	52.825	78.307	54.881	62.627	88.993	29.167	62.5	8.3333	39.239	3137	1872	388	103	7	15	2	111
ADL-Rundle-6	48.465	41.286	58.667	2068	2941	1457											
ADL-Rundle-6	3525	5009	167	24													

4.5 Analysis

The implementation of advanced methods in the object tracking system has led to significant improvements in key performance metrics, demonstrating the efficacy of these upgrades. These enhancements are particularly evident in the chosen metrics: HOTA (Higher Order Tracking Accuracy), IDF1 (ID F1 Score), IDSW (ID Switches), MT (Mostly Tracked), PT (Partially Tracked), and ML (Mostly Lost).

While the overall upgrades in the object tracking system have led to significant improvements, it's important to note a specific nuance regarding the use of the Kalman filter. Contrary to expectations, the integration of the Kalman filter consistently resulted in inferior performance compared to scenarios where it was not utilized. This outcome could be attributed to several factors specific to the nature of the videos being analyzed or the calibration of the Kalman filter's parameters.